

METHOD AND APPARATUS FOR UNIQUELY AND AUTHORITATIVELY IDENTIFYING TANGIBLE OBJECTS

FIELD OF THE INVENTION

5 The present invention relates to digital data processing devices, and in particular to identification and authentication using digital data processing technology.

BACKGROUND OF THE INVENTION

10 In some far distant past era, the rights of people over property were protected by a community, which both recognized its individual members and what belonged to them, either individually or collectively, as part of the common knowledge of the community itself. As the world became more complex, more populous, and more impersonal, these ancient forms of property protection became impractical, and forms more appropriate to the needs of changing societies were developed. At some point, the number of individuals in a community, or the
15 number of sheep, goats, or other items of property, multiplied to where it was difficult for the unaided memory to keep track of the various property relations. Man therefore conceived a method of identifying property to aid the memory, and to discourage interlopers: he branded each animal with a mark which was unique to the animal's owner.

20 From this primitive beginning, a great variety of techniques for protecting property relationships has developed. For example, recording systems, typically maintained by some governmental agency, have been used to record property relationships; serial numbers have been embedded in manufactured goods; special techniques have been used for printing currency, which makes it difficult to counterfeit; and code words or passwords have been used to limit access to physical areas or information. The list is almost endless.

In general, each new development in the art of identification and authentication is responsive to some perceived threat to existing relationships. As those who would infringe the rights of others become increasingly more sophisticated in their techniques, it is necessary to develop improved protection mechanisms. In other words, an "arms race" of sorts exists, in which the stability of the social order depends on continuing improvement in property protection methods and systems.

The complexity of the modern world has gone beyond the need to identify and authenticate items of property. The identity of an individual himself, once the common knowledge of a community, and later established by the memory of a subset of individual members of the community, is now usually established by reference to an identity document, such as a driver's license or passport. But like any other tangible object, an identity document can be stolen, copied, altered, or counterfeited. Thus, the same problems involved in identifying and authenticating a tangible object are now present in the case of identifying individuals as well.

The latter half of the twentieth century has been witness to a phenomenon known as the information revolution. While the information revolution is a historical development broader in scope than any one event or machine, no single device has come to represent the information revolution more than the digital electronic computer. The development of computer systems has surely been a revolution. Each year, the declining prices and expanding capabilities of computers and other digital technology cause them to be used in new and varied applications.

The development of digital computers and related digital data devices has dramatically increased the scope of the aforementioned arms race. The range of property to be protected, tangible and intangible, has greatly increased at the same time that digital technology provides imaginative new capabilities to infringers. On the other hand, digital technology opens up

vast potential resources for property protection, which have as yet been tapped to only a small degree.

Property protection has various aspects, among them being identification and authentication. These concepts overlap to a considerable degree, but it is useful to bear in mind the different purposes of identification and authentication. Identification refers to techniques whereby some object is distinguished from other similar objects, often for purposes of establishing ownership of the object in question. Identification is often unique, i.e., an identification associated with an object is unique in the universe of objects of the same type. However, identification need not necessarily be unique. Authentication refers to techniques for establishing that the attributes of an object are real and have not been forged. Both of these concepts may be applied to the identities and attributes of individuals as well as objects. For example, is an individual really who he claims to be (a question of identification), and does he have authority to perform some action (a question of authorization).

Many conventional techniques exist which use digital data devices for purpose of identification and/or authorization. For example, serial numbers can be encoded in read-only memory devices, or device attributes can be stored in on-line servers. The use of so-called smart cards is an outstanding example of digital technology employed as both an identification and authorization device (in this case, usually for an individual). But many gaps or shortcomings exist in protection mechanisms using conventional identification and/or authorization techniques.

One recurring problem with conventional techniques is a design trade-off between security and freedom of action. Naturally, key data should be protected from unauthorized copying or alteration, and sometimes from unauthorized observation as well. Conceptually, such data can be made secure if it is unalterable and unreadable. Unfortunately, such a requirement places impractical restrictions on the use of the data. There is usually a need to read data for various purposes, and often a need to alter it under controlled circumstances.

There are many ways in the digital world to alter data, but any capability to alter data opens the door to the possibility that unscrupulous parties will do just that to their own advantage.

A need exists for improved techniques for property protection, and in particular, improved techniques for using digital data technology to identify and authenticate tangible
5 objects (including documents used for the purpose of identifying/authenticating individuals).

SUMMARY OF THE INVENTION

A protected tangible object includes a digital data protection system having a small processor with the capability to decode data using a public/private key encryption algorithm,
10 and a permanent, secure storage accessible only through the processor. The digital data protection system is herein referred to as a "smart chip", it being understood that it is not necessarily required that all protection system elements be contained on a single chip. A substantially unique public and private identity key pair are generated according to such an encryption algorithm, the identity private key being stored in the permanent, secure storage
15 such that it can not be read outside the chip. The identity of the protection system is verified by performing a data transformation of test data according to the public/private key encryption algorithm using the identity public key, by requesting the protection system to perform a complementary transformation using the identity private key, and comparing twice transformed data with original data.

It is possible to use the identity public key directly as an identifier or index to
20 additional information, but in the preferred embodiment, an issuing entity generates a descriptor, the descriptor containing the identity public key, attribute data concerning the object (or person, if the object is intended as an identifying document), and a digital signature. In this preferred embodiment, the digital signature is generated by enciphering ("signing") a
25 derivation of the identity public key and the attribute data according to a public/private key encryption algorithm (which may or may not be the same algorithm as used for the identity

public/private key pair), using a signature private key and a signature public key pair (these being different from the identity public/private key pair). The signature private key is known only to the issuing entity, and is not stored on the smart chip.

5 In the preferred embodiment, the descriptor data is stored in a storage area on the smart chip itself, from which it is read out at the request of any requesting device. Thus, the descriptor data itself is not intended to be protected from being read. Alternatively, it is possible to store a reference to at least some of the descriptor data on the smart chip, the reference being used to access a data location on a separate medium. In this case, the smart chip may contain only the reference, or may contain some of the descriptor data along with
10 the reference. As a further alternative, the descriptor data may be stored on a separate storage area on the protected object, but external to the smart chip. The choice of alternative may depend on the application, the volume of descriptor data, and other factors.

15 The authenticity of the descriptor data can be verified by applying the signature public key to the digital signature using the decryption algorithm, and comparing the result to the appropriate derivation of the descriptor data. The identity of the object can be verified by encrypting a random number with the public identity key, requesting the smart chip to decipher it and return the deciphered value using its identity private key, and comparing the returned value to the original random number. Alternatively, the identity of the object can be
20 verified by requesting the smart chip to encrypt a random number using its identity private key, decrypting the encrypted random number using the identity public key, and comparing the decrypted value with the original random number.

25 In the preferred embodiment, the digital signature is generated by creating a hash of all data bits in the identity public key and attribute data portions of the descriptor, using any of various known hash algorithms, such as SHA-1 or MD5. This hash will change if any bits are modified in the original data. The hash is then encrypted using the signature private key.

It is preferred that the issuer of the smart chip maintain no record of the identity private keys it creates, and that the private keys be stored only in the smart chips themselves. If it is desirable to maintain a record of identifiers and owners, it is sufficient to record the identity public key and owner data in a database. In this manner, even if the issuer's security is compromised, the identity private keys remain secret, making duplication of previously issued smart chips nearly impossible.

The use of identity and signature keys as described herein are intended to prevent unauthorized copying or alteration of the attribute data, and do not necessarily prevent reading of the data. In many applications, it is desirable that the attribute data be in plain text to facilitate reading, and this may be done consistent with the present invention. In other applications, it may be desirable to restrict visibility of the attribute data, in which case the attribute data itself may be encrypted according to any of various encryption algorithms, also consistent with the present invention.

The preferred embodiment of the present invention thus achieves a significant level of security while permitting an issuing entity to easily update the attributes of a protected object. The descriptor data can be easily accessed, but it can not be readily used without the smart chip for which it is intended, and can not be readily altered without the private signature key in the hands of the issuing entity. Updating descriptor data can be accomplished by transmitting the old descriptor to the issuing entity, deriving the public identity key from the old descriptor, modifying the attribute data, generating a new digital signature using the new data and the private signature key, and transmitting the new descriptor back to the user. It will be observed that this process does not necessarily require any level of security during transmission, so transmission may be accomplished over the Internet or any other readily available means.

In a first exemplary use, the smart chip is embedded in a cellular telephone. The smart chip stores what is effectively a unique identifier for the phone (i.e., the identity private/public

key pair). The descriptor stored in the smart chip includes attribute data identifying the user, e.g., the user's telephone number. When a user dials out, the cell phone automatically transmits its descriptor data to the service provider. The service provider verifies the digital signature, encrypts a random number using the identity public key in the descriptor, and transmits the encrypted random number back to the cell phone, requesting that it decrypt the number. If the cell phone correctly decrypts the random number, its identity is verified. The service provider then verifies the contents of the descriptor, and if verified, the requested connection is made, and the call is billed to the user. If the cell phone can not decrypt the random number or the descriptor can not be verified, the call is terminated. Such a protection system makes it difficult for potential thieves to trap a cell phone identification, and use the identification to fraudulently bill the user for other telephone calls.

In a second exemplary use, the smart chip is embedded in certain significant parts of an automobile or other complex machine. A central machine controller has a list of all parts which should be present, and their respective identifiers. The central controller may itself include a smart chip, the list of identifiers being part of the descriptor data of the central controller's smart chip. Alternatively or additionally, an identifier of the central controller may be part of the descriptor data in the smart chip of each part. When the automobile or other machine is powered on, the central controller polls all the significant parts, and can determine whether the identity of each part matches the appropriate identifier. The action taken at that point may vary with the circumstances. The controller may issue a warning message to the user, or may shut itself off after a predetermined time, or may simply shut down immediately.

In a third exemplary use, the smart chip is embedded in a personal identity document, for example, a passport. The descriptor data includes identifying information about the subject, such as name, date of birth, sex, height, weight, and so on. The descriptor data also includes a digitized photographic image of the subject. The descriptor data may further include other more sophisticated forms of personal identifying information, such as

fingerprints, retina or iris scan, voice record, DNA markers, and so on. Such an identity document would be extremely difficult to copy or alter. Furthermore, properly used in conjunction with sophisticated identifying equipment, it would make identification almost foolproof. For example, an identification apparatus at a port of entry could verify the identity of each individual by first verifying the identify of the passport and authenticity of descriptor data as described above, and then comparing at least some of the descriptor data to the bearer of the passport (e.g., performing an iris scan of the bearer's iris, and comparing the scan to the descriptor data).

In a fourth exemplary use, the smart chip is embedded in a satellite television receiving system. The smart chip stores what is effectively a unique identifier for the system (i.e., the identity private/public key pair). The descriptor stored in the smart chip includes attribute data identifying the television broadcasts which the user is entitled to receive, and may include the descrambling keys needed to receive the broadcasts. An on-board digital controller in the satellite receiver system obtains the descriptor from the smart chip, uses the signature public key to verify that the descriptor data has not been altered, and uses the identity public key to verify that the descriptor data belongs with the physical smart chip within the receiver system (thus preventing some unscrupulous person from merely copying a valid descriptor from a different smart chip). If the customer wishes to change his viewing options, the issuing entity can easily generate another descriptor as described above.

The details of the present invention, both as to its structure and operation, can best be understood in reference to the accompanying drawings, in which like reference numerals refer to like parts, and in which:

BRIEF DESCRIPTION OF THE DRAWING

Fig. 1 is a high-level block diagram of a smart chip for use in identifying or authenticating a tangible object, according to the preferred embodiment of the present invention.

5 Figs. 2A and 2B are high-level illustrations of two alternative general environments for using a smart chip, according to the preferred embodiment.

Fig. 3 is a conceptual illustration of the major software components of a digital device which uses a smart chip to identify or authenticate an object, according to the preferred embodiment.

10 Figs. 4A and 4B are collectively a high-level flow diagram of the process of extracting and verifying data from a smart chip, according to the preferred embodiment.

Figs. 5A, 5B, and 5C are collectively a high-level flow diagram of the process of updating descriptor data associated with a smart chip, according to the preferred embodiment.

15 Fig. 6 is a high-level block diagram of the major components of such a cellular telephone, according to an exemplary first use of the preferred embodiment of the present invention.

Fig. 7 is a high-level flow diagram illustrating the operation of the control program of a cellular telephone in conjunction with a service provider communication station, according to an exemplary first use of the preferred embodiment of the present invention.

20 Fig. 8 illustrates the major components of a motor vehicle parts network, according to an exemplary second use of the preferred embodiment of the present invention.

Fig. 9 is a high-level flow diagram illustrating the part verification process at power-on of a motor vehicle, according to an exemplary second use of the preferred embodiment of the present invention..

25 Fig. 10 represents the major components and usage of a passport control station, according to an exemplary third use of the preferred embodiment of the present invention.

Fig. 11 is a high-level flow diagram illustrating the process of verifying and screening the bearer of a passport at a passport control station, according to an exemplary third use of the preferred embodiment of the present invention.

5 Fig. 12 is a high-level block diagram of the major components of a satellite television receiving system, according to an exemplary fourth use of the preferred embodiment of the present invention.

Fig. 13 is a high-level flow diagram illustrating the operation of the control program of a satellite television receiving system, according to an exemplary fourth use of the preferred embodiment of the present invention.

10

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Public/Private Key Encryption Overview

15 In accordance with the present invention, one or more public/private key encryption algorithms are used. Various such algorithms exist today and are known in the art. Any such algorithm or algorithms, now known or hereafter developed, could be used consistent with the present invention, the exact choice of algorithm depending on various design considerations.

20 These algorithms have the characteristic that they perform a one-way data transformation using a key. By "one-way" it is meant that it is not algorithmically feasible to perform a reverse data transformation to obtain the original data, having only the transformed data and the key used to construct it. However, it is possible to perform a reverse data transformation with a second key. The two keys therefore form a pair, with which it is possible to perform a complete transformation and reverse transformation of data.

25 The one-way nature of the data transformations necessarily implies that it is not algorithmically feasible to obtain one key from the other, for if it were, it would be possible

to perform the reverse data transformation by simply generating the other key. However, since there must be a way to generate the two keys initially a further characteristic of a public/private key encryption algorithm is that it is possible to algorithmically generate a pair of keys. The key generation algorithm itself requires some input data (a seed), so that it will not always generate the same key pair. Typically, this is accomplished by seeding a key generation algorithm with a random number.

Public/private key encryption algorithms are commonly used to encrypt and decrypt data (including encryption keys for data) being transmitted over or stored in an unsecure medium. For most data of significant size, it is common to encrypt the data using a fast algorithm, such as DES, which provides more efficient transformation than public/private key encryption, and to encrypt/decrypt only the encryption key for the fast algorithm using a public/private key encryption algorithm. In this environment, the initial transformation is an encryption of the data, and the reverse transformation a decryption. The key used to encrypt the data is called a public key, because it need not be protected, and multiple copies of it can be distributed. A person in possession of the public key can only encrypt, and can not decrypt, data. The private key is available only to persons having a need to decrypt the data, usually a single person. Thus, almost anyone might be able to generate an encrypted version of some data, but only the holder of the private key can read such data.

These algorithms can, however, be used in a different mode, in which the private key is used to generate the encrypted version of the data, while the public key is used to decrypt it. Obviously, such a usage is not intended to protect the data from being read, since anyone with the public key can decrypt and read the data. This usage is intended to authenticate data. The encrypted version of the data is referred to as a digital signature, and it normally accompanies a plain text version of the data to authenticate the plain text version. Since only the originator of the data has the private key, only the originator can generate an encrypted version, i.e., generate a digital signature. By performing the reverse transformation (decryption) with the public key and comparing the decrypted data to the plain text version,

anyone can verify that the plain text version is the same as what the originator had encrypted in the digital signature. Thus data is protected from *alteration*, as opposed to *access*, by third parties. Unless the data being authenticated by a digital signature is very brief, it is common to generate a digital signature from a hashing of the data, since an encryption of the entire data may result in unreasonably large digital signatures.

As used above, “algorithmically feasible” refers to the capability to perform an act (in this case, a data transformation) using a known algorithm which is guaranteed to, or which will in general, complete in a reasonable time. In the mathematical sense, it is possible to perform the reverse transformation by systematically generating every possible combination of original data, transforming it with the original key, and comparing the result to the transformed data, until a match is found. However, for most encrypted transformations of even a relatively small amount of data, it can readily be demonstrated that the fastest computers available today would take longer than the projected age of the universe to complete such a task, and hence this approach can not be said to be “algorithmically feasible”. Similarly, it may be possible with substantial effort under certain circumstances to perform a reverse transformation using heuristics, if it is known or guessed that certain patterns exist in the original data. No encryption algorithm can claim to forever defeat the ingenuity of potential codebreakers, but encryption techniques can make it enormously difficult to get at protected information, so difficult that the cost of doing so typically outweighs the potential benefits.

General Description of the Preferred Embodiments

Referring to the Drawing, wherein like numbers denote like parts throughout the several views, Fig. 1 is a high-level block diagram of a smart chip 100 for use in identifying or authenticating a tangible object, according to the preferred embodiment. Smart chip 100 is preferably implemented as an integrated circuit on a single “chip” substrate. As shown in Fig 1, smart chip 100 comprises a programmable processor 101, a non-volatile program

memory 102, a non-volatile internal hidden memory 103, and a general purpose memory 104, all of which are coupled to an internal bus 110 for transferring data between the memories and the processor. Smart chip 100 may optionally contain a non-volatile external data memory 105, also coupled to bus 110. Processor 101 is coupled to an external interface 111 for receiving and transmitting data to and from the chip. It will be observed that the memories are preferably arranged so that external data communications always pass through processor 101; while this arrangement is not necessarily the most efficient for transferring large volumes of data, it is normally expected that the volume of data crossing the external interface 111 is low, and this arrangement provides some level of security from unauthorized access to data in the memories.

Processor 101 is preferably a simple programmable general-purpose processor, which executes a set of instructions of a control program to control the operation of smart chip 100. Because smart chip 100 is not expected to be used for large scale intensive data processing like a general-purpose digital computer, processor 101 may be slower, have a more limited instruction set, number of I/O lines, registers, etc., than a typical central processing unit found in a general purpose computer. Program memory 102 contains the executable instructions which form the control program 112 executing in processor 101. Program memory 102 must be non-volatile, i.e., the contents must be maintained in the absence of power to smart chip 100. Since it is not expected that control program 112 will be altered, and is in fact preferred that the program can not be altered, program memory 102 is preferably permanently encoded with data at manufacture, and thereafter acts as a non-writable read-only memory.

Hidden memory 103 is used for storing the identity private key 113, the use of which is explained in greater detail herein. Hidden memory must also be non-volatile. Memory 103 is called hidden because one requirement of the memory is that its contents can not be read outside smart chip 100. Since processor 101 needs to access hidden memory 103 for certain operations, control program 112 is responsible for enforcing this restriction. Hidden memory 103 is not necessarily physically constructed as read-only memory, and could be, e.g., a flash

memory. Depending on the application, it may be possible to write to hidden memory externally, as long as one can never read hidden memory externally. In practice, smart chip 100 may be mass-produced by permanently forming the same control program 112 in program memories 102 of all smart chips of a given model, and subsequently inserting individual identity private keys 113 into the hidden memories by sending a predetermined series of special commands to the smart chip. It may be desirable to prevent subsequent alteration of the hidden memory. This may be accomplished, e.g., by severing a fusible link needed to program the memory after the same has been programmed with a value. However, in most applications it is believed that no security exposure exists from the ability of anyone to write a new private key to the smart chip as long as the existing private key can not be read, because the data descriptor containing the original identity public key is signed with a signature private key which is unknown outside the issuer, and changing the identity private key will invalidate the descriptor.

General purpose memory 104 is a read/write random access memory (RAM) used by processor 101 to store temporary values while performing operations. It is not necessary that general purpose memory 104 be non-volatile.

External data memory 105 is an optional non-volatile memory used for storing data available to be read externally. In particular, external data memory 105 may be used to store descriptor data 114, i.e., an identity public key, attribute data, and a digital signature, as more fully explained herein. For reasons which will be explained below, it is not necessary to prevent descriptor data from being read externally. Therefore, in a further alternative embodiment, descriptor data 115 could be stored in an external data memory 106 located outside smart chip 100. External data memory 105 could alternatively store a portion of the descriptor data, and/or could store an index to a location containing the descriptor data, the remaining descriptor data being stored in another memory location. Memory 105 may store additional data to be read externally. External data memory 105 and/or 106, whichever is utilized, is preferably a read/write memory to allow updating of data. It is expected that in

most applications data will be updated infrequently, and therefore a memory technology which allows only a limited number of write operations is acceptable.

Fig. 1 shows four different memory modules within smart chip 100. However, it should be understood that Fig. 1 is intended as a conceptual representation for purposes of showing the different functional parts of memory in smart chip 101. In fact, some of the memories shown in Fig. 1 might be combined in a single memory structure, the different modules shown in Fig. 1 being merely different address locations in the memory structure.

The present invention may utilize any of various smart chip implementations known in the art. For example, chips of the AT90SC family by Atmel Corporation may be used, it being understood that this represents but one example of various alternative smart chip design, now existing or hereafter developed, which may be used.

In general, a tangible object protected by a smart chip in accordance with the present invention may be a non-electronic device to which the smart chip is attached, or may itself be a digital electronic device in which the smart chip is embedded. These two general implementations are illustrated at a high level as Figs. 2A and 2B, respectively.

As shown in Fig. 2A, smart chip 100 is attached to tangible object 201. In this variation, tangible object 201 is itself a non-electronic object, i.e., it contains no digital electronic logic or power which is coupled to chip 100. Example of such non-electronic objects are passports, bank charge cards, certain automobile parts, etc. The smart chip is intended to be plugged into a coupling 218 of a digital electronic device 210 in order to be used. The digital electronic device may e.g., comprise a processor 211, a random access memory 212, input/output interfaces 213-214 for handling input and output through terminals, printers, networks, etc., data storage such as a rotating magnetic disk drive 215, and a communications bus 216 for communicating data among the various components. In this configuration, smart chip 100 receives power and communicates data through coupling

218. Smart chip 100 may be plugged into coupling 218 on a transient basis (e.g., in the case of a tangible object 201 which is a passport which is verified at a passport control station), or may be plugged into coupling 218 on a more or less continuous basis (e.g., in the case of an automotive part which is plugged into an on-board central control computer of an automobile).

Although a physical coupling which is capable of transmitting electrical power as well as data is shown in Fig. 2A, it will be appreciated that other forms of data interface would be possible, particularly if the tangible object includes an independent power source such as a battery for the smart chip. E.g., a battery-powered smart chip could communicate with a digital electronic device via an infrared link or other form of remote transmission.

As shown in Fig. 2B, smart chip 100 is an integral part of digital data device 220, which is the protected tangible object. Digital data device 220 may contain, e.g., processor 221, memory 222, input/output interfaces 223-224, and data storage 225, which communicate with each other through communications bus 226.

In either the configuration of Fig. 2A or Fig. 2B, smart chip 100 is permanently attached to the respective protected object (i.e., object 201, or digital data system 220) in such a manner that it is not practical to remove it and attach it to other like objects. Almost any chip can be removed with either sufficient force and/or by using the reverse of complex manufacturing processes, but for this purpose it is sufficient if removal is either highly likely to destroy the chip, prohibitively expensive, or impractical for some other reason. Various adhesives or attachment processes are available which would create a bond between the smart chip 100 and the protected object such that removal is likely to destroy the chip or otherwise impractical, the exact choice of means usually depending on the type of object being protected.

The representations of Fig. 2A and 2B are intended as an exemplary simplified representations, it being understood that many variations in digital device configuration are possible in addition to those mentioned here. The number and type of attached devices may vary considerably. Digital devices 210 or 220 may be any of various models of single-user computer systems known as "personal computers". Devices 210 or 220 may also be larger multi-user computer systems such as an IBM Enterprise system or an IBM AS/400 system. While a single interactive display, keyboard and pointing device are shown in each of Figs. 2A and 2B, such a multiuser system may support the attachment of multiple terminals, each having a display and input means. Such multiple terminals may have their own processors and memory, so that various functions could be distributed between the processor of the mainframe computer system and the processors of the terminals. For example, multiple interactive workstations could be attached to a mainframe computer system, the mainframe storing a database of service provider and client data, while user interactions are provided through the interactive workstations. Alternatively, the present invention could be implemented using a client-server computing model in which a central server or servers of the service provider store data or perform certain tasks on behalf of the service provider's requesting terminals which act as clients (not to be confused with clients of the service provider). Finally, digital device 210 or 220 need not be a general purpose computer system, but could be any of various special-purpose digital devices. For example, the digital device could be an on-board automotive computer control system, a satellite television receiving system, a robotic control system, etc. In this case, the number and type of I/O devices or other devices are typically more limited. For example, such systems often lack a general purpose display and keyboard for user interface, although limited function interfaces may be provided. The present invention should therefore not be construed as limited to any particular type or configuration of digital device.

Fig. 3 is a conceptual illustration of the major software components of a digital device which uses a smart chip to identify or authenticate an object. For illustrative purposes, the various software components are shown in Fig. 3 as being within memory 212 of Fig. 2A, it

being understood that this could also represent the memory 222 of Fig. 2B. As shown in Fig. 3, operating system 301 provides various low-level software functions, such as device interfaces, management of memory pages, management of windowing interfaces, management of multiple tasks, etc. as is well-known in the art. In addition, operating system 301 contains verification and extraction function 302 for verifying and extracting data from smart chip 100, as explained in greater detail herein. Verification and extraction function 302 is shown in Fig. 3 as part of the operating system; however, it will be appreciated that it could be part of an application program or a stand-alone application, and in either case typically would not be part of operating system 301. Memory 212 further includes various application programs 303-304, which may use data contained in data files 311-313. Application programs may further use data, particularly attribute data, associated with the protected object. For example, if the protected object is a bank card, an application program invokes the verification and extraction operating system function 302 to receive attribute data relating the bearer's identifying characteristics, credit, account numbers, and so forth, and may use this data to verify the bearer's identity (e.g., comparing a signature of the bearer to a pre-recorded facsimile signature in the attribute data), verify credit, charge the correct account, etc. Additionally, applications 303-304 may include applications which are unrelated to the use of smart chip 100 and do not use data therefrom, such as, word processing, spreadsheet, electronic calendar, accounting, graphics, or any of thousands of other possible applications. Applications 303-304 utilize data contained in database files 311-313.

While a certain number of application programs, files or other entities are shown in Fig. 3, it will be understood that these are shown for purposes of illustration only, and that the actual number and type of such entities may vary. Additionally, while the software components of Fig. 3 are shown conceptually as residing in memory, it will be understood that in general the memory of a digital data system will be too small to hold all programs and data simultaneously, and that information is typically stored in data storage 215 or 225, comprising one or more mass storage devices such as rotating magnetic disk drives, and that the information is paged into memory by the operating system as required.

Figs.4A and 4B (collectively referred to herein as Fig. 4) constitute a high-level flow diagram of the process of extracting and verifying data from the smart chip, performed by the extract and verify function 302 in conjunction with smart chip 100, according to the preferred embodiment. In this embodiment, a digital device extracts and verifies information from a smart chip. This process involves an exchange of information between the digital device and the smart chip. Fig. 4 illustrates this process by a generally vertical dashed line dividing Fig. 4 into a left portion and a right portion, the left portion representing steps performed by the digital device, and the right portion representing steps performed by the smart chip.

The extracting and verifying function 302 is shown as part of the operating system of the digital device, although it may alternatively be a separate application or a part thereof. Some other program (e.g., an application program or a control program) calls the extracting and verifying function 302 in order to extract data from a smart chip and verify the data. The verifying function is intended to verify that (a) the data extracted belongs to the physical smart chip from which it was taken, and (b) the data extracted has not been altered in an unauthorized manner. If verification completes successfully, the extract and verify function 302 returns the extracted data.

The identity public key of the smart chip is verified by generating test data (also referred to as "challenge data"), performing a data transformation according to the pre-defined public/private key encryption algorithm using the identity public key, requesting the smart chip to perform the complementary data transformation according to the pre-defined public/private key encryption algorithm using its hidden identity private key, and comparing data which has undergone both transformations to the original test data. Test data is preferably randomly generated data, although this is not strictly required in all environments. It will be observed that it is equally possible to perform the identity public key transformation first, followed by the complementary transformation using the identity private key, as it is to perform the identity private key transformation first, followed by a complementary transformation using the identity public key. In the embodiments described below, the

identity public key transformation (referred to as an encryption using the identity public key) is performed first, followed by the complementary transformation (referred to as a decryption using the identity private key). The consistent use of a single convention in the detailed description herein (encryption using the identity public key, followed by decryption using the identity private key) is intended to promote ease of understanding the various embodiments of the present invention, and is not intended to limit the scope of the invention to this particular sequence. Consistent with the present invention, it is possible to generate test data, request the smart chip to encrypt the test data using its identity private key, decrypt the encrypted test data using the identity public key, and compare the result to the original test data.

As shown in Fig. 4, upon invoking the extract and verify function 302, the digital device requests the descriptor from the smart chip (step 401). Upon receiving the request, the smart chip's processor 101 retrieves the descriptor from its external data storage 105 (step 402), and returns the descriptor data across interface 111 to the digital device.

In the preferred embodiment herein described, all descriptor data is stored in the smart chip itself in an appropriate external data storage area 105. This option is generally preferred where the volume of descriptor data is not large. However, it must be recognized that there will be situations in which the volume of descriptor data is so large that it can not be conveniently stored in chip 100, or there are other reasons for storing it elsewhere. In such cases, many possible variations exist for storing all or portions of the descriptor data elsewhere. For example, the data stored in smart chip 100 may include only a public identity key, digital signature, and a database index. Upon the return of partial descriptor data from the smart chip, the digital device may be required to access a database (which may be external to the digital device) using the index in order to obtain complete descriptor data. Furthermore, the descriptor data may be stored in one or more additional memory chips (external to smart chip 100) which are permanently attached to a protected tangible object. The additional steps required to retrieve full descriptor data from other locations, such as

databases or storage external to smart chip 100, are not shown in Fig. 4.

5 The extract and verify function 302 of the digital device obtains the signature public key used for verifying the digital signature within the descriptor data (step 404). The signature public key, being intentionally public, need not be protected. It is normally intended that all protected tangible objects of a particular model or type or from a particular manufacturer use the same signature public key. Therefore, the signature public key might be contained in a data file in the digital device itself, or could even be hard-coded in the extract and verify function itself. Alternatively, the signature public key might be retrieved from an external database.

10 The extract and verify function 302 then decrypts the digital signature contained within the descriptor data using the signature public key and a pre-defined public/private key encryption/decryption algorithm (step 405). The public/private key encryption/decryption algorithm will herein be referred to as the signature encryption algorithm. Extracting fields such as the digital signature from the descriptor data requires that the data have a pre-determined format; e.g., pre-defined delimiters can be used for identifying the signature or other data within the descriptor data.

20 The digital signature represents an encrypted derivation of certain descriptor data. Specifically, the source data for the digital signature is the descriptor data other than the digital signature, i.e., the identity public key and the attribute data. This source is hashed using a known hash function and encrypted using the signature private key and the signature encryption algorithm. The encrypted version thus generated is the digital signature. In order to preserve data integrity, the signature private key should be accessible only to the entity which is authorized to generate the descriptor data, referred to herein as the issuer. Thus, all digital signatures are generated by the issuer using a set of descriptor data.

Hashing is performed because, in general, the amount of descriptor data would create an excessively large digital signature if encrypted in its entirety, yet it is desirable to protect all of the descriptor data. The hash can be constructed using any of various known hashing algorithms, provided that the algorithm to be used is defined to the digital device. Either an MD5 or an SHA-1, which create 16-byte and 20-byte hashes, respectively, are appropriate hashing algorithms, it being understood that other hashing algorithms now known or hereafter developed could also be used. It should further be understood that, in the broadest sense, a digital signature is constructed from some derivation of the descriptor data in order to provide an indication whether the descriptor data has been altered, but the derivation is not necessarily a hash of the descriptor data. For example, if the descriptor data to be protected is sufficiently brief, the digital signature might be simply an encryption of the descriptor data itself.

The extract and verify function 302 then constructs a hash of the descriptor data other than the digital signature, i.e., the descriptor data consisting of the identity public key and the attribute data, which was used as a source for generating the digital signature by the issuer (step 406). The decrypted digital signature generated at step 405 is then compared to the hashed descriptor data generated at step 406. If the data match, then the descriptor data read from the smart chip is the same as that which was used to generate the digital signature, indicating that the data has not been altered in an unauthorized manner. In this case, the "Y" branch is taken from step 407. If the data do not match, then the descriptor data has somehow been altered after the signature was generated or the descriptor data was signed by a different private key, and the "N" branch is taken from step 407.

Assuming proper authentication of the descriptor data (the "Y" branch from step 407), the extract and verify function then determines whether the data itself matches the physical smart chip from which it was removed. This procedure is necessary because, without it, it may be possible to simply replicate a descriptor (digital signature and all) from a validly issued smart chip, and thus replicate any desirable attributes from the validly issued chip. In

order to do this, the function extracts the identity public key from the descriptor data (step 408), and generates random data (step 409). The random data is then encrypted using the identity public key and a pre-defined public/private key encryption/decryption algorithm, herein referred to as the identity encryption algorithm (step 410). The identity encryption
5 algorithm could be the same as the signature encryption algorithm, but need not be.

The digital device then requests the smart chip to decrypt the encrypted random data generated at step 410, sending a command across the interface to the smart chip (step 411). Upon receiving the command, the smart chip's processor 101 accesses the identity private key in hidden storage 103 (step 412), and decrypts the received encrypted random data using the
10 identity private key and the identity encryption algorithm (step 413). The decrypted data is then returned to the digital device (step 414). The control program for the smart chip does not permit processor 101 to return the identity private key.

Upon receiving the returned decrypted data, the extract and verify function in the digital device compares the returned data with the randomly generated data before encryption
15 (step 415). If the data matches, then the public key in the descriptor data belongs to the physical smart chip from which it was extracted; if not, the data may have been copied from another source and can not be considered reliable.

In the case of a data match, the "Y" branch is taken from step 415, and the attribute data is extracted from the descriptor and made available in some manner to the calling
20 program (step 416). This could be accomplished, e.g., by returning the attribute data itself, or by writing the attribute data to a record or file accessible to the calling program. The extract and verify function then returns with an indication of successful completion (step 417).

If the data does not match at either step 407 or step 415, the extract and verify function returns with an appropriate error indication (step 420). The form of error indication
25 and action to be taken by the calling program will vary depending upon the application. In

some cases, the extract and verify function may return the attribute data, and the calling program may use it notwithstanding that it is considered unreliable, the calling program providing an appropriate indication such as a warning message to a user or to some other system or system component. In other cases, the failure of the extract and verify routine is
5 cause for aborting a requested operation or shutting down a system or subsystem.

It will be appreciated that the order of certain steps may be altered, and that in particular it would alternatively be possible to first verify the identity of the smart chip, and then verify the digital signature of the attribute data.

Depending on the application, the digital data device may under some circumstances
10 wish to alter attribute data stored in or associated with smart chip 100. In accordance with the present invention, attribute data can be updated in an authorized manner under reasonably secure conditions, and without undue logistical difficulty. Attribute data in a smart chip is preferably updated by performing a cooperative procedure involving the smart chip itself, the digital device into which the smart chip is plugged, and a remote central server under the
15 control of the issuer. The digital device communicates with the remote server using any appropriate data communications medium. Since there is no requirement that the communications medium be secure, the Internet or the telephone network are appropriate communications media, although other media could be used. This procedure thus avoids the need to physically transport the protected object and its attached smart chip to the issuer for
20 updating.

Figs. 5A, 5B, and 5C (collectively referred to herein as Fig. 5) constitute a high-level flowchart illustrating a preferred method for updating descriptor data (which includes attribute data) associated with a smart chip. Fig. 5 illustrates the division of process steps between the smart chip, the locally attached digital device, and the remote server, by two
25 generally vertical dashed lines dividing Fig. 5 into a left portion, a middle portion and a right portion, the left portion representing steps performed by the smart chip, the middle portion

representing the steps performed by the local digital device, and the right portion representing steps performed by the remote server under control of the issuer.

As shown in Fig. 5, the local digital device (with coupled smart chip) reads and optionally verifies the existing attribute information in the smart chip. This process is described above and shown in Fig. 4, it being represented in Fig. 5 as step 501. Specifically, the local digital device and smart chip perform steps 401-403 to read the descriptor from the smart chip. Optionally, the local digital device also verifies the descriptor as shown and described above with respect to steps 404-420. Because the central server also verifies the descriptor as described below, it is not strictly necessary that the local digital device do so in this case. If the verification process fails, the action taken may depend on the application and reason for failure. E.g., if the verification process fails because the smart chip is unable to successfully decrypt random data using its identity private key, then it is unlikely that any update can be performed; on the other hand, if the identity private key is verified but the digital signature does not match the current attribute data, the application may choose to simply request new attribute data and a new digital signature.

An update request relating to the attribute data is then formulated at the local device (step 502). The update request may itself contain the new attribute data, or may be a request that the server generate the new attribute data according to some parameters specified by the local device. The local device may solicit user input in generating an update request. When the request is generated, it is transmitted to the remote server (step 503). Generally, this request data transmitted to the remote server will include the entire existing descriptor to enable the central server to perform full verification of the existing descriptor contents, but it may include only the identity public key, or it may include the identity public key together with a portion of existing descriptor data.

Upon receipt of the update request, the central server verifies the identity of the device (smart chip) to which the request pertains. This process is shown in Fig. 5 as steps 504-513, and is similar to the process performed by a local device in verifying the identity of the smart chip, described above. The server first extracts the ID public key from the descriptor data which was sent with the update request (step 504). The server generates random data to be used for verifying identity (step 505). The random data is then encrypted using the identity public key and the pre-defined public/private key identity encryption algorithm (step 506). The server then transmits the encrypted random data to the local device, requesting that it be decrypted by the smart chip (step 507).

Upon receipt of the encrypted random data generated at step 506, the local device requests the smart chip to decrypt the data, sending a command across the interface to the smart chip (step 508). Upon receiving the command, the smart chip's processor 101 accesses the identity private key in hidden storage 103 (step 509), and decrypts the received encrypted random data using the identity private key and the identity encryption algorithm (step 510). The decrypted data is then returned to the local device (step 511), which transmits it to the central server (step 512).

Upon receiving the returned decrypted data, the central server compares the returned data with the original randomly generated data (step 513). If the data matches, then the public key in the descriptor data belongs to the physical smart chip making the update request, i.e., the identity of the smart chip has been verified. In this case, the "Y" branch is taken from step 513, and update processing continues. If the data does not match, then all bets are off. Any number of things, either fraudulent or accidental, might have caused a mismatch, but it is impossible to update descriptor data if the central server can not verify the identity of the smart chip. Accordingly, the "N" branch is taken from step 513, and descriptor data is not updated.

If the “Y” branch is taken from step 513, the server optionally verifies the contents of the existing descriptor, shown as steps 515-518. As shown in Fig. 5, the “Y” branch from step 513 diverges into two paths, one going through steps 515-518, and the other by-passing these steps to go directly to step 520. Verification of the descriptor contents is optional because, in at least some applications, the descriptor will be completely overwritten by a new descriptor, and the central server doesn’t care what was in the old attribute data. For example, in the motor vehicle network application described herein, the server may simply overwrite the old data when a part is legitimately exchanged, without verifying the contents. However, in general it is assumed that the central server will verify the descriptor contents. In many cases the identity of the user is contained in the attribute data, and it is necessary to verify the user’s identity before proceeding further. In others, any unauthorized alterations to attribute data may indicate fraud or some other circumstance requiring investigation.

The verification of the descriptor data by the server is similar to verification of such data by the local device, described above with respect to steps 404-407 of Fig. 4. The server verifies the descriptor data by retrieving the signature public key, a copy of which would preferably be kept by the server (step 515), and decrypting the digital signature within the descriptor using the signature public key and the pre-defined public/private key signature encryption algorithm (step 516). The server then derives a hash or other pre-defined function of the source data for the digital signature, i.e., the identity public key and the attribute data (step 517), as explained above with respect to step 406.

If the decrypted digital signature matches the hashed data (step 518), then the descriptor contents have been verified. In this case, the “Y” branch is taken from step 518, and update processing continues. If the data do not match, then the descriptor data has somehow been altered after the signature was generated, and the “N” branch is taken from step 518.

Having verified the identity of the smart chip, and if necessary, the contents of the descriptor data, the server determines whether the request will be approved, and if so, approves it (step 520). In general, it will be desirable that the central server maintain control over the update process, which implies an approval process. In some cases the approval may
5 in fact be automatic. More typically, approval will require access to one or more databases to verify information, such as payment of a fee, sufficient credit in an account, an absence of a report of theft or fraud, and so forth. If the request is not approved for any such reason, the “N” branch is taken from step 520; otherwise the update processing continues via the “Y” branch from step 520.

10 The server generates new attribute data, as necessary (step 521). In some cases, the new data will be provided entirely by the requesting local device with the request (subject to approval by the server), making this step unnecessary. In other cases, the server must generate it. For example, if the request is to add an additional channel to the reception capability of a satellite receiver, the attribute data may include an encryption key for the
15 additional channel. In this case, the local device will probably not have the key, and can only request it using some appropriate channel identifier. The server will fetch the key from a database.

The server then accesses its signature private key (step 522), generates a hash of the identity public key together with the new attribute data according to a pre-defined hashing
20 algorithm (step 523), and encrypts the hash using the private signature key and the signature encryption algorithm (step 524). The encrypted hash created at step 524 represents the digital signature.

The new descriptor, comprising the identity public key, new attribute data, and newly generated digital signature, is then transmitted back to the local digital device using the
25 communications medium (step 525). Upon receipt of the new descriptor data, the local digital device requests the smart chip to store the data (step 526), and the smart chip stores

the descriptor data in its appropriate place (step 527), completing the process of updating descriptor data. This may mean requesting the smart chip to store it in its external data storage 105, or may mean storing it in any other storage used for the descriptor data. Since the new descriptor can only be used/verified by the smart chip whose identity private key
5 corresponds to the identity public key in the descriptor data, there is no concern that the descriptor data can be intercepted and used on a different device from that for which it was intended.

In the event that an update request fails any of the tests described above with respect to steps 513, 518 or 520, the central server has an appropriate mechanism for handling the
10 cause of rejection, shown generically as step 530. This could vary considerably depending on the application, and may amount to anything from a simple rejection message to the requestor to alerting law enforcement authorities for appropriate action.

In the embodiment described above, it has been generally assumed that it is not necessary to protect attribute data from being read, and only necessary to protect it from
15 being altered. In many cases, this will be true, and it will be more convenient to simply store attribute data in plain text form, available to be read by any device having an appropriate interface. However, it will be recognized that in some applications it may be desirable to protect the attribute data from being read as well as to protect it from being altered. In these circumstances, the additional level of protection can be accomplished by encrypting the
20 attribute data. Attribute data could be encrypted using a third pair of keys (i.e., keys different from either the signature key pair or the identity key pair) and possibly an encryption algorithm different from either used for the signature or the identity. But it is also possible to use the identity key pair and corresponding identity encryption algorithm, thus avoiding the need for additional keys. Specifically, when the issuer initially generates or subsequently
25 updates attribute data, all or a portion of the data is encrypted using the identity public key and the identity encryption algorithm. The digital signature is then generated by hashing the identity public key and this encrypted form of the attribute data, followed by an encryption

using the signature private key, as previously described. In order to read the attribute data, verification steps 401-417 are performed as described earlier. Additionally, the digital device passes the encrypted attribute data to the smart chip, requesting the smart chip to decrypt the same. The smart chip accesses its identity private key, decrypts the encrypted attribute data, and returns it to the digital device. The decryption of attribute data could be performed before or after the identity of the smart chip is verified, or before or after the digital signature is verified.

First Exemplary Use: Cellular Telephone Device

A first exemplary use of a smart chip protection mechanism is now described. In this use, the smart chip is embedded in a cellular telephone (which is the "digital device"). The smart chip stores a form of unique identifier for the telephone (i.e., the identity private/public key pair). However, in general the identifiers used by the telephone industry will be the telephone number, and may also include the user's name, account number, and so forth. This data is preferably stored as attribute data within the descriptor, which is stored in the smart chip and protected from alteration by the digital signature.

The major components of such a telephone are shown in Fig. 6. Physically, the telephone is typically portable, receiving power from a battery, and having no fixed location. However, in some areas of the world, cellular telephones are used in fixed locations in preference to so-called land lines, and the present invention is not limited to portable cellular telephones. As is well known, a cellular telephone subscriber either owns or leases a phone, and pays a fee to a service provider to obtain access to a telephone network through the service provider's communications station. The service provider's communications station is typically a fixed local tower which communicates with cell phones within a defined area, although it could be a satellite or some other form of communications link, now known or hereafter developed.

Telephone 600 contains a transceiver 601 for transmitting signals to and receiving signals from the service provider's communications station. Signals include voice transmissions, which may or may not be encrypted, and further include control signals necessary for cellular system operation. Telephone 600 further contains control processor 602, which is a programmable processor for executing a control program; read-only memory 603 containing a control program which executes on control processor 602 to control the operation of telephone 600; dynamic random access memory 604 for storing state information and other temporary data, or for buffering; smart chip 605; I/O interface 606; and internal data bus 607 for communicating data among the various components of the system. I/O interface 606 drives the local I/O devices, specifically a speaker 610 for emitting sound to the user, a microphone 611 for receiving the user's voice, and a keypad 612 for receiving user commands, such as dialing telephone numbers and so forth. I/O interface may additionally drive other I/O devices (not shown), such as status lights and the like. Battery 608 supplies power to the various components of phone 600. It will be recognized that Fig. 6 is intended as a simplified representation of a cell phone, and that an actual system may contain other or additional components or be configured differently.

One of the practical problems in implementing any cellular telephone system is the need to identify the originator of a communication. By its nature, the communications link between the user's telephone and the service provider's communications station can be used by anyone. In order to bill the proper party, the originator must identify himself to the service provider's system. Although passwords or other identification devices may be employed, there is a risk that a user's identity may be stolen. Once stolen, the service provider's system can not distinguish between legitimate calls originating from the user, or pirated calls originating from a cell phone programmed to claim the same user identity.

In this first exemplary use, the identity of the cell phone is verified by the service provider's system with every originating telephone call by a process similar to that described above for verifying smart chip data. I.e., the service provider generates random data,

encrypts it with the identity public key of the user, sends the encrypted data to the cell phone for decryption in the cell phone's smart chip, and receives the decrypted data back. The descriptor data is then verified by decrypting the digital signature and comparing it to a hash of the descriptor data. The process of verifying the cell phone's identity and descriptor data is similar to the process shown and described above with respect to Fig. 5 when a descriptor is to be updated.

Fig. 7 is a high-level flowchart illustrating the operation of the control program executing in processor 602 of telephone 600, in conjunction with a service provider communication station. As shown in Fig. 7, upon power-on, the control program reads and verifies the attribute data in smart chip 605 by performing the steps described herein and illustrated in Fig. 4; this process is represented in Fig. 7 as step 701. The control program then initializes its dynamic state variables in RAM 604, and in particular, stores the descriptor in RAM 604. The control program then enters a wait state (step 702), in which it waits for some event. An event may be the initiation of some action by the user (e.g., pressing a button or opening a cover to indicate that the user is ready to take some action), or it may be the receipt of an incoming telephone call from the service provider communication station. Depending on the design of the cell phone, there may be multiple types of wait state, including a state in which the phone is unable to receive incoming calls, and a state in which the phone is available to receive incoming calls.

If the user wishes to initiate a call, then the "Y" branch is taken from step 703. If any other operation is required, the "N" branch is taken from step 703, the required operation is performed (step 704), and the control program returns to the wait state after performing the operation. Such an operation could include, e.g., receipt of an incoming call, or could include any of various operations supported by the cell phone.

If the user is initiating a telephone call from the cell phone, the phone transmits a call request to the service provider's communications station, using transceiver 601 (step 705).

The call request will include the telephone number to be called and any additional data needed by the service provider, and will also include the descriptor. Upon receipt of the call request, the service provider extracts the identity public key from the descriptor, generates random data to be used for verifying identity, and encrypts the random data using the identity public key and the pre-defined public/private key identity encryption algorithm (step 706). The communications station then transmits the encrypted random data to the cell phone, requesting that it be decrypted (step 707).

Upon receipt of the encrypted random data, the cell phone's controller requests smart chip 605 to decrypt the data, using its identity private key and the identity encryption algorithm, which it does (step 708). The cell phone then returns the decrypted data to the communications station (step 709).

Upon receiving the returned decrypted data, the communications station compares the returned data with the original randomly generated data (step 710). If the data matches, then the public key has been verified. In this case, the "Y" branch is taken from step 710, and update processing continues. If the data does not match, the "N" branch is taken from step 710, and the requested connection can not be made.

If the "Y" branch is taken from step 710, the communications station verifies the contents of the descriptor (specifically, the telephone number of the originating cell phone, which is contained in the descriptor). This is accomplished by retrieving the signature public key, a copy of which would preferably be kept by the communications station; decrypting the digital signature within the descriptor using the signature public key and the pre-defined public/private key signature encryption algorithm; and deriving a hash or other pre-defined function of the source data for the digital signature, i.e., the identity public key and the attribute data. This process is illustrated in Fig. 7 as step 711, and is the same as that used by a local device to verify the descriptor, as explained above with respect to steps 404-406.

If the decrypted digital signature matches the hashed data (step 712), then the descriptor contents have been verified. In this case, the "Y" branch is taken from step 712, and the communications station makes the call connection (step 713). Upon being connected, the cell phone handles the call until completion (step 714), and then returns to the wait state. If the data do not match, then the descriptor data has somehow been altered after the signature was generated or a different private key was used to generate the signature, and the "N" branch is taken from step 712.

In either case where the identity key or the descriptor can not be verified, the call is not completed, and the communications station will take appropriate error recovery action (step 715), which will typically include an error message to the cell phone.

It will be observed that it would be possible to update a descriptor in a cell phone (e.g., to change the telephone number of a physical phone, the capabilities associated with the phone, etc.) using the process described above and illustrated in Fig. 5. A subscriber may initiate such a request, as when requesting a change in some capability of the phone. But it is also possible that in such a case, it will be the service provider that initiates the update request. For example, a service provider may store information concerning itself (access numbers and so forth) in the attribute data of its subscribers. If any of this information changes, the service provider may initiate an update of the descriptors in the telephones of its subscribers. The service provider could do this by polling the subscribing telephones, and asking each in turn to transmit its descriptor for updating. Alternatively, the service provider might have the descriptors of its subscribers already stored in a database, so that polling for the descriptors is not necessary.

The operation of the cell phone in receiving a telephone call is not explained in detail above. In the preferred first exemplary use, the identity of the originating telephone is always verified as described herein. For added protection, it would be possible to verify the identity of the cell phone by using the smart chip, as herein described with respect to steps 705-712,

when calls are received as well as when the cell phone initiates calls. I.e., upon receipt of an incoming call message from the service provider's communication station, the cell phone would respond by transmitting its descriptor to the communication station, and the communication station and cell phone would then perform steps 706-713 described above to verify the identity of the receiving phone and establish the connection.

Fig. 7 is intended as a simplified representation of cell phone operation, and in particular, the steps taken to verify the identity of a caller when a cell phone initiates a call. It will be appreciated that the operation of a cell phone may be considerably more sophisticated, and many functions not explained in detail here, either now known or hereafter developed, may be performed in addition to initiating and receiving telephone calls. For example, the cell phone may have the capability to store incoming messages for later playback, may have the capability to access the Internet to obtain e-mail or other data, or may have additional capabilities. Such additional capabilities do not negate the need for verification of the user's identity, and in fact may increase the need for such verification, and call for additional data in the descriptor. For example, if an Internet account is to be accessed from a cell phone, the descriptor stored in the smart chip could contain a log-on identifier.

In the description of cellular telephone operation provided above, the telephone service provider is treated as a single monolithic entity for simplicity of illustration. In fact, the telephone network is a global network of numerous interconnected service providers. When a cellular telephone user initiates a call within the local service area of his own service provider, it is possible that all of the essential service provider functions described above will be performed by a single entity. However, in many cases the call will initiate within the zone of another service provider, and data may be exchanged among multiple service providers to provide the function above described. E.g., the service provider for the zone receiving the call request may act as a conduit, passing the request to one or more other service providers to eventually reach the user's service provider, and the user's service provider may then generate random test data and so forth to verify the identity of the caller. As used herein,

unless otherwise clearly limited by the context, "service provider" includes a single service provider or a network of service providers which perform the applicable function.

It will be noted that it would be alternatively possible to use the subscriber's identity public key as his user identifier, and that in this case, it would only be necessary to verify the identity public key by decrypting random data (steps 705-710), without verifying descriptor data. This public key, once verified, could be used to index a database containing any additional user information required. However, in the preferred embodiment, the user's identifying data is contained in the descriptor data. Although this requires additional verification steps, it is considered a more flexible and practical approach. Public keys are cumbersome as identifiers. In the telephone industry, the user will typically be identified primarily by his telephone number, which is unique in all the world. Such a universal identifier may facilitate shared provision of telephone services by multiple service providers in different local areas.

Second Exemplary Use: Motor Vehicle Network

A second exemplary use of a smart chip protection mechanism is now described. In this use, a plurality of smart chips are embedded in respective replaceable parts of a motor vehicle, the use being applicable in addition to other complex machines. One of the protected parts is an on-board computer/controller ("digital device") for controlling various electronic systems of the motor vehicle. The remaining protected parts do not necessarily have any electronic components other than the attached smart chip. Such protected parts may include, e.g., the engine block, unibody or chassis, transmission, radio/cassette/CD player, fenders, etc. The on-board computer/controller has a list of all parts which should be present, and their respective identifiers, which might be, e.g., vehicle identification numbers, serial numbers, or other identifiers. The controller itself preferably includes a smart chip, the list of part identifiers being contained in the attribute data on the controller's smart chip. Additionally, an identifier of the central controller or a vehicle identifier is preferably part of the descriptor data in the smart chip of each part. When the vehicle is powered on, the

controller polls all the parts in its list to determine whether the identity of each part matches the list. Such a system discourages both the theft of parts, particularly through so-called automotive "chop-shops", and the inappropriate or substandard substitution of parts.

5 Fig. 8 illustrates the major components of such a motor vehicle network 800. All the components illustrated in Fig. 8 are mounted within a motor vehicle. As shown in Fig. 8, a central controller 801 (which is in reality a small, special-purpose computer system) contains programmable processor 802; read-only memory 803 storing a control program for executing on the processor to control the operation of the various vehicular systems; dynamic random
10 access memory 804 for storing state information and other temporary data; smart chip 805; smart chip interface driver 806 for communicating with other smart chips in the vehicle network; various other interfaces 807 for communicating with sensors, control mechanisms, and other devices (not shown); and internal controller bus 808 for communicating data among the various components of controller 801. Smart chip interface couples controller
15 801 to signal and power wires leading to smart chips 810-812 attached to various vehicular components. In this embodiment, smart chips 810-812 are physically attached but not electrically coupled to their respective component parts, the smart chips receiving power externally and communicating all data with controller 801. While a fender, a body and an engine are depicted for illustration in Fig. 8, it will be understood that the actual number and
20 type of components may vary.

Preferably, smart chip 805 includes in its attribute data a list of part identifiers for each part having an attached smart chip. Smart chip 805 also includes the vehicle identification number as part of its attribute data. Additionally, each smart chip 810-812 contains as part of its attribute data a global part identifier, which could be the part identifier
25 of controller 801, or a vehicle identification number. Thus, the relationships of part identifiers to the central controller are recorded in two directions. It would alternatively be possible to record identifiers in only one direction (e.g., only the controller contains the list of part identifiers, and the parts themselves do not contain an identifier pointing back to the

controller or the vehicle identifier), but recording in both directions provides increased capability to track parts which end up in the wrong place.

For example, as one alternative, the list of parts could be maintained as part of the attribute data of the central controller's smart chip 805. A permanent identifier would be necessary in each part. However, the identifier would not necessarily have to be a smart chip having attribute data as described herein. It would be sufficient that a smart chip in the part has a permanent identifier (such as the identity public key) which can be verified by the controller. It would not even be strictly necessary that the parts (other than the central controller) contain a smart chip. The part may contain a simple permanent, non-volatile and non-writable read-only memory which records a unique identifier. This alternative is considerably less secure, since there is a potential for anyone to copy such an unprotected identifier.

Preferably, the control program verifies the part identifiers when the vehicle is powered-on. Because vehicles are not left continuously running, and it is unlikely that parts would be replaced while the vehicle is running, a verification performed once at power-on is considered sufficient. Fig. 9 is a high-level flow diagram illustrating the part verification process at power-on.

As shown in Fig. 9, upon power-on the control program executing in processor 802 initializes state variables and creates a table of parts to be verified in RAM (step 901), the table at this point not containing unique part identifiers. The control program then reads and verifies the descriptor data in smart chip 805, meaning that it performs the process shown and described with respect to Fig. 4, this process being represented in Fig. 9 as step 902.

The attribute data in smart chip 805 includes the identifier of each respective part in the table of parts to be verified. If the verification performed at step 902 completed normally, the "Y" branch is taken from step 903, and the part identifiers obtained from smart

chip 805 are loaded into the table of parts to be verified (step 904). If verification failed, the "N" branch is taken from step 903, and appropriate failure action is performed at step 910.

5 The control program then sequentially verifies each part in the table of parts. At step 905, the control program determines whether any parts remain in the table to be verified. If so, the "Y" branch is taken, causing the control program to read and verify the descriptor data in the smart chip (i.e., one of 810-812) corresponding to the part to be verified. This verification process is shown and described above with respect to Fig. 4, and is represented in Fig. 9 as step 906.

10 The attribute data in each smart chip 810-812 includes the identifier of the respective part as well as the controller's identifier or other global identifier. If the verification performed at step 906 completed normally, the "Y" branch is taken from step 907. The controller then compares the part identifier read from the smart chip attached to the respective part with the part identifier in the table in RAM, which was earlier read from controller smart chip 805. If these part identifiers are identical, the "Y" branch is taken from step 908. The control program then compares the controller or other global identifier read from the smart chip attached to the respective part with the controller or other global identifier read from controller smart chip 805. If these part identifiers are identical, the "Y" branch is taken from step 909; at this point, the part has been verified, and the control program continues to step 905 to verify the next part. When all parts have been verified, the "N" branch is taken from step 905, and the part verification process is complete.

25 If the "N" branch is taken from any of steps 903, 907, 908 or 909, the part in question can not be verified. Where the part is the controller (step 903), further verification is aborted after an appropriate action is taken (step 910), since the list of part identifiers can not be assumed accurate. Where the part is any other part, appropriate action is taken (step 911), and the control program continues to verify the remaining parts. The "appropriate action" to be taken may depend on a variety of engineering, legal, or marketing factors. In general,

in the case of a motor vehicle, it will be undesirable to disable the vehicle simply because a part verification failed. There may be circumstances, e.g., where emergency repairs made it necessary to replace one part with another, which the controller does not recognize. Therefore, it is assumed that appropriate action will generally include some form of advisory or warning message to the user. For example, the inability to verify the part may simply mean that the part is a replacement part manufactured by a non-recognized supplier. In this case, an informational message may be appropriate. It may also mean that the failure is logged in a non-volatile area of controller memory, so that if the vehicle is later serviced, it can be retrieved. Furthermore, if the vehicle is equipped with data transmission capabilities for communicating with remote service facilities of a vehicle manufacturer or service provider, the vehicle may automatically send a message to the remote service facility. Finally, if the verification failure is considered sufficiently serious, the controller may advise the user that it is setting a timer (e.g., 30 days) within which the user must service the vehicle, at the end of which time the controller will disable the vehicle if the problem is not corrected.

It will be recognized that there will be many legitimate circumstances where either new or used parts are used to replace existing parts in a motor vehicle, and that in these circumstances there should be a means for updating attribute information so that future parts verifications can complete normally without error messages to the user or other annoying action. In general, this process is shown in Fig. 5, described earlier, with some minor modification..

In order to facilitate updating smart chip part data, the data is typically read from the vehicle at an authorized repair facility, using a computer and special data coupling which connects to the on-board vehicle controller. If a part is to be replaced, the replacement part is installed first. The repair facility's computer makes the controller read descriptor data both for itself, and for all parts in the vehicle (step 501). The repair facility's computer then formulates the update request and transmits it to a central server under the control of the

vehicle manufacturer or its agent (steps 502 and 503). Normally, any such request coming from an authorized source will be approved by the manufacturer, but the manufacturer may check records of stolen vehicles before approval. Additionally, the manufacturer may maintain its own databases of update requests, part identifiers and corresponding vehicles, in case it discovers later that the part was from a stolen vehicle. The manufacturer then generates new descriptors for the controller and for each part in the vehicle (or only for selected replacement parts) as described above with reference to steps 505-508 in Fig. 5. The descriptors are then transmitted back to the repair facility (step 526), downloaded from the repair facility's computer into the on-board controller, and ultimately stored in the controller's smart chip 805 and the smart chip(s) of the affected parts (step 527).

The second exemplary use as herein described need not be limited to motor vehicles, and could be extended to other types of complex devices to be protected. For example, construction machinery, aircraft, manufacturing installations, etc. Furthermore, while one of the major concerns in the case of automobiles is use of stolen parts, there are additional reasons justifying the use of the present invention in complex machinery: assurance of proper version level of a particular part, control of additional useful data such as date of installation, which may be used to trigger reminders that the part needs servicing or replacement (particularly useful in the aircraft industry), or verification that the part comes from a particular source.

Third Exemplary Use: Personal Identity Document

A third exemplary use of a smart chip protection mechanism is now described. In this use, a smart chips is embedded in a personal identity document, which is preferably a passport. The passport has no internal power source or data processing capability apart from the smart chip itself, and is intended to be plugged into an external computer at a port of entry or similar passport control station in order to use the smart chip. The smart chip's attribute data includes standard identifying information about the subject, such as name, date of birth, sex, height, weight, and so on. The descriptor data preferably also includes a

digitized photographic image of the subject. The descriptor data preferably further includes at least one other more sophisticated forms of personal identifying information, which typically requires the use of a computer to verify an individual's identity. In the exemplary embodiment, data from an iris scan is used, it being understood that fingerprint, retina scan, voice scan, DNA markers, and so on could alternatively be used.

Fig.10 is a representation of the major components of the third exemplary use, in which an individual's identity and passport are verified at a passport control station. As shown in Fig. 10, a passport control station 1001 (which is a small workstation connected to a central database on one or more computers through a network) comprises a general purpose computer having an attached iris scanner 1010 for scanning the iris of a passport bearer, scanner interface 1011, and smart chip interface/coupling 1012 for connecting to a smart chip 1003 embedded in the passport 1002 of a bearer. The computer of the passport control station includes programmable processor 1013; dynamic random access memory 1014 for storing programs and data for use by processor 1013; mass data storage 1015, such as one or more rotating magnetic disk drive storage devices, network interface 1016 for communicating with one or more server systems (not shown) through a network, and terminal interface 1017 for communicating with a terminal 1018, which includes a data display and input means such as a keyboard, mouse, and the like. An internal bus 1019 supports communication between the various elements of the passport control station computer.

Smart chip 1003 is permanently attached to the passport in a manner such that separation is likely to permanently destroy the chip or leave obvious visible evidence. Passport 1002 contains no power source, and when carried around by the bearer, smart chip 1003 is not powered. When smart chip 1003 is plugged into smart chip interface 1012 of the passport control station, the interface supplies power to the chip and handles data communication between the chip and the passport control station's computer.

In its ultimate application, the smart chip and digitally stored data are the actual passport, and the thing or document holding the chip is just a carrier. However, it will be recognized that it is not practical to introduce digital passport interface devices in every local passport control station in the world simultaneously. Therefore, it is expected that the paper version of the passport will duplicate much of the digitally stored data in the chip, and this paper version will be used by stations which are unable to perform the full range of verification described herein. It is further possible that some passport control stations will have limited digital processing capability, without the full range herein described. E.g., a station may be able to communicate with a smart chip and generate a random number to verify the identity private key, but unable to perform an iris scan to match iris characteristics stored in the descriptor with the passport bearer.

A local passport screening program 1020 executes in processor 1013, and is stored in memory 1014; Fig. 11 represents the program in memory 1020, it being understood that portions of the program may be in memory while the full program is in storage 1015, and is paged into memory as needed. This local program preferably communicates with a central server as each passport bearer is examined. The process of verifying and screening the bearer of a passport is depicted at a high level in Fig. 11. It will be understood that, depending on the function performed at a passport control station, other and additional actions may be taken which are not represented in Fig. 11. For example, luggage may be searched, the bearer may be questioned, etc.

As shown in Fig. 11, upon approaching the passport control station, the bearer or the operator plugs the passport's smart chip 1003 into the smart chip interface 1012 (step 1101), causing the smart chip to power on and initialize itself. The local screening program 1020 then reads and verifies the smart chip data from the smart chip, following the procedure shown in Fig. 4 and described in greater detail above; this process is represented in Fig. 11 as step 1102. If the verification succeeds without any error message, the "Y" branch is taken from step 1103. The screening program then displays the data it read from the smart chip to

an operator on the display screen of terminal 1018 (step 1104). The operator may compare personal identifying data with the physical characteristics of the bearer. In addition, identifying data may indicate to the operator the type of questions to be asked or other important matters.

5 At approximately the same time, the local screening program 1020 transmits at least some identifying data to a central server system via network interface 1016 (step 1105). Identifying data is transmitted to the central server at step 1105 in order to permit the server to access databases of information which would not necessarily be contained on the passport's smart chip. As one obvious example, although by no means the only one, any of
10 various law enforcement agencies may have an interest in the bearer, and it may be appropriate to detain the bearer as a result. Additionally, any of various government agencies may wish to maintain records of the departures and arrivals of individuals, even if no action is to be taken to interfere with these events.

15 While the local passport control station is awaiting a response from the central server, the bearer is instructed to put an eye to the iris scanner 1010 for identification purposes, and a scan is taken (step 1106). The iris scanner scans the iris of the bearer's eye, and this information is temporarily stored in memory 1014. As is known in the art of iris scanning identification, the iris of an eye contains numerous identifying features, which vary from individual to individual, and even between the two eyes of the same individual. A sufficient
20 number of these features can be extracted such that the odds against any two random individuals having the same iris features are astronomical, and therefore, for all practical purposes, an enumeration of iris features can be considered unique. The attribute data stored in smart chip 1003 includes an enumeration of the iris features of the bearer. The screening program 1020 processes the iris scan and extracts an enumeration of the iris features of the
25 bearer from the scan (step 1107). This enumeration is compared with the data stored on the smart chip (step 1108) to provide an extremely effective verification of the identity of the bearer.

If the iris features match, the “Y” branch is taken from step 1108, and the screening program 1020 waits for a response from the central server, if necessary (step 1109). Upon receipt of a response, the response is displayed to the operator for any necessary action (step 1110). In the normal case, this will merely mean that the bearer is authorized to pass through the passport control station, although there will obviously be exceptions in which other instructions are given to the operator.

If the “N” branch is taken from either step 1103 or 1104, then either the data on the passport can not be deemed reliable, or the passport must be assumed to belong to another individual, and appropriate action is taken (step 1111). In either case, it can be assumed that the bearer will have a lot of explaining to do.

In the third exemplary use, it is probable that the passport issuer will want to maintain strict control over the issuance and modification of attribute data in the passport. Because different jurisdictions are unlikely to share signature private keys, and such sharing would create a greater security exposure, passports issued by different jurisdictions will probably use digital signatures encoded with different signature private keys. The passport control station would need a list of signature public keys of all jurisdictions issuing smart chip passports. The passport’s smart chip would contain some form of jurisdictional identifier, from which the passport screening program could retrieve the appropriate key.

In the third exemplary use, remote updating of attribute data as shown in Fig. 5 and described above may be considered undesirable. On the other hand, it may be desirable to record entries and exits from different jurisdictions or other information in the attribute data. This could be accomplished by using multiple descriptors, each having a digital signature encoded with a different signature private key according to the jurisdiction which placed the descriptor in the passport. Thus, in addition to screening and providing information to the operator, the central server could provide an updated descriptor for the jurisdiction of the passport control station, which is returned to the passport control station and loaded into the

passport's smart chip as a record of having entered or left the jurisdiction.

Fourth Exemplary Use: Satellite (or Cable) Television Receiving System

5 A fourth exemplary use of a smart chip protection mechanism is now described. In this use, the smart chip is embedded in a satellite television receiving system (which is the "digital device"). In general, the description herein applies equally well to cable television receiving systems. The smart chip stores what is effectively a unique identifier for the system (i.e., the identity private/public key pair). The descriptor stored in the smart chip includes attribute data identifying the television broadcasts which the user is entitled to receive, and may include additional user identifiers such as a name or account number. 10 Specifically, it is preferred that the attribute data include the descrambling information needed to receive authorized selective channels, i.e., those to which a user subscribes.

15 The major components of such a satellite receiving system are shown in Fig. 12. Physically, the receiving system is typically located in the home of a subscriber. The receiving system may be owned or leased by the subscriber, but in any case the subscriber pays for the privilege of receiving the service. The receiving system 1200 is coupled to and receives signals from satellite receiving dish 1201, and provides unscrambled video input to television display 1202. Receiving system 1200 includes a television signal transmission interface 1210 for receiving the incoming signal from the satellite dish and driving an output video signal to the television display 1202; control processor 1212, which is a programmable 20 processor for executing a control program; read-only memory 1213 containing a control program which executes on control processor 1212 to control the operation of system 1200; dynamic random access memory 1214 for storing state information and other temporary data; smart chip 1215; and internal data bus 1216 for communicating data among the various components of the system. It will be recognized that Fig. 12 is intended as a simplified 25 representation of a satellite receiver system, and that an actual system may contain other or additional components or be configured differently.

In this fourth exemplary use, the control program executing in control processor 1212 periodically polls smart chip 1215 to extract the current attribute data. Attribute data includes descrambling information for the channels the subscriber is entitled to receive. This information may be nothing more than a bit mask, in which one bit is assigned to each channel, indicating whether or not the subscriber is entitled to receive it. Alternatively, it may include descrambling keys for each channel the subscriber is entitled to receive, the keys being used to descramble the received television signal. For added security, descrambling keys could be encrypted within the descriptor using the identity public key of the smart chip; the control program would request the smart chip to decrypt the encrypted descrambling keys, using the identity private key of the smart chip. Whatever its form, the descrambling information is stored in a channel table 1217 in RAM 1214. When a subscriber selects a particular channel to receive, the control program accesses the channel table to determine the subscriber's authorization. If necessary the appropriate descrambling key will be obtained from the table and used to descramble the signal.

Typically, channels are scrambled simply to make it difficult for an unauthorized person to view the signal, and not to prevent all access to data, as in the case of encryption of sensitive data. The bandwidth of a television signal may make full encryption impractical, and therefore simpler techniques are generally used to scramble a signal. However, the present invention is not limited to any particular technique for scrambling the television signal, and could apply to any technique now existing or hereafter developed, including the encryption of a signal.

Fig. 13 is a high-level flowchart illustrating the operation of the control program executing in processor 1212 of receiving system 1200. As shown in Fig. 13, upon power-on, the control program reads and verifies the attribute data in smart chip 1215 by performing the steps described herein and illustrated in Fig. 4; this process is represented in Fig. 13 as step 1301. The control program then initializes its dynamic state variables in RAM 1214, and in particular, constructs a channel table 1217 which contains a list of channels and

corresponding descrambling data read from the attribute data in smart chip 1215. The data for the currently selected or default channel is initialized as the current channel data. If necessary, descrambling data is decrypted before entering it in table 1217. The current channel data is used to set the state of the signal transmission interface 1210, i.e., the transmission interface transmits the selected channel signal to the television display 1202, descrambling the signal as indicated by the descrambling data. The current channel data may, e.g., be held in a register (not shown) within signal interface 1210.

The control program then enters an operating state wherein it repetitively checks for various conditions and responds, represented generally as steps 1303-1315. While in this operating state, the signal interface 1210 continuously receives incoming signals from receiver dish 1201 on a selected channel, descrambles the signal if so indicated by the descrambling data for the current channel, and transmits the resultant signal to television display 1202. If a user makes a new channel selection (step 1303), the control program gets the channel descrambling data corresponding to the newly selected channel from table 1217, and loads this as the current channel data (step 1304). The control program also changes the selected channel for the interface 1210 to receive.

In addition to video signals, receiver system 1200 may receive commands broadcast by satellite. Such commands could be broadcast over a separate command channel, or could be interleaved in the video signal of any programming channel. Commands may be of a sort which are intended for all receiving systems (e.g., a clock set signal), or may be intended for a single receiving system, in which case the command will carry a unique system identifier. Upon receiving a command (step 1309), the control program determines whether the command is intended for system 1200 (step 1310); if not, it is ignored. If the command is intended for system 1200, the control program executes the command (step 1311). Among other commands, the received command could contain the unique identifier for system 1200, and could direct it to store an updated descriptor in smart chip 1215. By this means, it is possible to receive updated attribute data, to be stored in smart chip 1215, via satellite.

Because only smart chip 1215 has the identity private key needed to decrypt the encryption keys, the updated descriptor is protected notwithstanding that it is broadcast via satellite to any receiving device.

5 The receiving system periodically reloads the smart chip data into RAM. When a timeout of the reloading period occurs (step 1312), the control program again reads and verifies smart chip data (step 1313), as described above and illustrated in detail in Fig. 4. It then overwrites the channel table and current channel data with the data read from the smart chip (step 1314). Periodic reloading prevents a subscriber from keeping obsolete
10 descrambling data or data to which he is no longer entitled in the channel table.

 If the subscriber selects power-off (step 1315), execution ends; otherwise, the control program returns to step 1303.

 Fig. 13 does not depict the action to be taken in the event that the verification function performed at either step 1301 or step 1313 fails, and this is a business decision for
15 the service provider. It will be assumed that, in general, the service provider will deny service (i.e., the receiver will shut itself off, or will cease descrambling of any scrambled channels) in this event. However, the service provider may wish to give the subscriber a warning that service will be discontinued if some action is not taken before a deadline. E.g., the subscriber may be required to return the receiver unit for replacement.

20 In the television industry, it is common for a subscriber to change his subscribing options, i.e., to change the channels he is authorized to receive. Such a change is often accompanied by a change in service fee. Additionally, a subscriber may pay to receive specific broadcasts. Alteration of the attribute data in smart chip 1215 can be made as
25 described herein without the need for service personnel to visit the subscriber's home, and without the need to physically bring the receiver unit into a service location.

In order to facilitate updating of subscriber choices, a modified version of the process depicted in Fig. 5 is used. Because there may be no practical way for receiver device 1200 to transmit data to the central server (satellite reception being one-way), steps 501-503 depicted in Fig. 5 are not performed. Instead, the service provider of satellite television services maintains a database of subscribers, corresponding identity public keys, device identifiers (such as serial numbers) of the receiving devices, customer subscription options, credit information, and the like. For security purposes, it is preferable that the service provider have no record of identity private keys. The service provider also maintains the signature private key, which must necessarily be maintained in secrecy. A customer wishing to change his subscription options initiates a request by telephone, mail, internet, or the like. The subscriber need not know the identity public key or other data relating to his receiver. The customer simply gives his name and desired subscription options to initiate the request.

The service provider's server then approves the request by accessing credit information and so forth, or receives human input approving the request (step 504). The server retrieves the subscriber's identity public key from its database, device identifier, channel encryption keys, and any other information necessary to construct new attribute data for the subscriber's receiving system (step 505). As part of constructing the attribute data, the channel descrambling information may be encrypted using the identity public key. Steps 506 through 508 are performed as described earlier. The complete updated descriptor is then transmitted to the subscriber's system by satellite broadcast (step 509), using the subscriber's identifier to specify the receiving system. The subscriber's system stores the new descriptor in its smart chip upon receipt of the broadcast data (step 510). Any receiving system other than the subscriber's system will ignore the updated descriptor (since it has a different identifier), and would be unable to use it in any case because it would lack the identity private key needed to verify the descriptor and decrypt the channel decryption keys.

Additional Variations

In general, the routines executed to implement the illustrated embodiments of the invention, whether implemented as part of an operating system or a specific application, program, object, module or sequence of instructions may be referred to herein as “computer programs” or simply “program”. The computer programs typically comprise instructions which, when read and executed by one or more processors in the devices or systems in a computer system consistent with the invention, cause those devices or systems to perform the steps necessary to execute steps or generate elements embodying the various aspects of the present invention. Moreover, while the invention has and hereinafter will be described in the context of fully functioning computer systems, the various embodiments of the invention are capable of being distributed as a program product in a variety of forms, and the invention applies equally regardless of the particular type of signal-bearing media used to actually carry out the distribution. Examples of signal-bearing media include, but are not limited to, recordable type media such as volatile and non-volatile memory devices, floppy disks, hard-disk drives, CD-ROM’s, DVD’s, magnetic tape, and transmission-type media such as digital and analog communications links, including wireless communications links. Examples of signal-bearing media are illustrated in Fig. 1 as program memory 102, in Fig. 2A as memory 212 and data storage device 215, and in Fig. 2B as memory 222 and data storage device 225.

In various illustrative uses described herein, it is desirable to assign some form of unique identifier (such as a serial number) to an object protected by a smart chip. Since the identity public key is considered unique for all practical purposes, it would be possible to use the key itself as an identifier, obviating the need for a separate identifier. However, in general the identifier may be used for other purposes (such as indexing information in a database, and therefore a separate (and usually shorter) identifier will often be desired.

It will be recognized that, given sufficient time and resources, it may be possible to read the identity private key from a smart chip. E.g., electrical probes of the chip, thermal

or timing analysis, or other means might conceivably be used to obtain an identity private key. However, the only thing a pirate can do with such a key is make an exact duplicate of the existing smart chip, including all its descriptor data. I.e., the service provider's security system itself remains essentially intact, and identity private keys of other users remain uncompromised. Once a service provider learns that a particular identity private key has been copied, the corresponding public key is simply deactivated in the service provider's database. Any subsequent attempt to use the key, e.g., to update the descriptor data, or perform a transaction requiring access to the service provider's database, will run across the deactivated notation in the database and cause the service provider to refuse service. Thus, if a pirate makes a large number of illicit copies, all of these are easily deactivated once their existence is known to the service provider, without affecting the validity or operation of other legitimate users.

As explained herein, the purpose of the digital signature is to be able to detect alteration of data. In various exemplary embodiments described herein, a digital signature is obtained by performing a hash function of the data to be protected (i.e., the identity public key and the attribute data), and then encrypting the result using a public/private key encryption algorithm and a signature private key which is preferably known only to the issuer. This technique provides a relatively high degree of security while still permitting attribute data to be altered by the issuer. However, other techniques for generating digital signatures may be used. For example, the data to be protected may be directly encrypted without hashing or other alteration (i.e., the identity function may be used in place of the hash function); a checksum or similar derivation of the data to be protected may be encrypted; etc. Additionally, there may be alternatives to encryption using the public/private key algorithms. As used herein, a digital signature may include any data which can be used to detect data alteration of itself or other data, either now known or hereafter developed.

In the preferred embodiment, random data is generated and encrypted as test data to verify the identity of smart chip devices. It will be understood, however, that in a strictly

mathematical sense it is impossible for a finite state machine such as a computer to generate truly "random" data. Various algorithms exist for generating data which approximates randomness, these typically being seeded with a variable, such as a system clock value, and any of these would generally be considered random. Furthermore, it may be sufficient to generate test data according to any of various algorithms or mechanisms which generate variable data which does not have an easily discernable pattern. Finally, in some environments, it may not matter whether patterns exist in the test data. E.g., if the test data is first encrypted by the smart chip using the identity private key, followed by decryption using the identity public key, it is assumed that the test data can be obtained by a snooper, and the test data merely needs to be different from previously used test data.

Although a specific embodiment of the invention has been disclosed along with certain alternatives, it will be recognized by those skilled in the art that additional variations in form and detail may be made within the scope of the following claims:

What is claimed is: